Лекция №3.
по курсу «Мобильное программирование»
часть 2

Москва 2019

Циклы

Вывод числа от 1 до 1000.

```
public class Loop {
    public static void main(String[] args) {
        int i = 1;
        while (i <= 1000) {
            System.out.println(i);
            i++;
        }
    }
}
```

Цикл с пост условием.

```
public class Loop {
    public static void main(String[] args) {
        int a = 0;
        do {
            System.out.println("Привет");
        } while (a > 0);
    }
}
```

Цикл с фиксированным количеством итераций

```
public class Loop {
    public static void main(String[] args) {
        for(int i = 0; i < 100; i++) {
            System.out.println(i);
        }
    }
}
```

break – прерывание цикла

МАССИВЫ

№	Объявление массива, Java-синтаксис	Примеры	Комментарий

1.	`dataType[] arrayName;`	`int[] myArray;`
		
		`Object[] arrayOfObjects;`
			Желательно объявлять массив именно таким способом, это Java-стиль

2.	`dataType arrayName[];`	`int myArray[];`
		
		`Object arrayOfObjects[];`
			Унаследованный от C/C++ способ объявления массивов, который работает и в Java

Как и любой другой объект, создать массив Java, то есть зарезервировать под него место в памяти, можно с помощью оператора **`new`**. Делается это так:

```
1  new typeOfArray [length];
```

Где **`typeOfArray`** — это тип массива, а **`length`** — его длина (то есть, количество ячеек), выраженная в целых числах (`int`). Однако здесь мы только выделили память под массив, но не связали созданный массив ни с какой объявленной ранее переменной. Обычно массив сначала объявляют, а потом создают, например:

```
1  int[] myArray; // объявление массива
2  myArray = new int[10]; // создание, то есть, выделение памяти для массива на 10 элементов типа int
```

МАССИВЫ

int[] myArray = new int[10];
получаем массив из десяти целых чисел, и, пока это не измениться в ходе программы, в каждой ячейке записан 0. массив с данными ссылочного типа, то по умолчанию в каждой ячейке записаны null

```
1  String[] seasons  = new String[4]; /* объявили и создали массив. Java выделила память под массив из 4
2
3  seasons[0] = "Winter"; /* в первую ячейку, то есть, в ячейку с нулевым номером мы записали строку Wint
4  seasons[1] = "Spring"; // проделываем ту же процедуру с ячейкой номер 1 (второй)
5  seasons[2] = "Summer"; // ...номер 2
6  seasons[3] = "Autumn"; // и с последней, номер 3
```

Теперь во всех четырёх ячейках нашего массива записаны названия сезонов. Инициализацию также можно провести по-другому, совместив с инициализацией и объявлением:

```
1  String[] seasons  = new String[] {"Winter", "Spring", "Summer", "Autumn"};
```

Более того, оператор `new` можно опустить:

```
1  String[] seasons  = {"Winter", "Spring", "Summer", "Autumn"};
```

МНОГОМЕРНЫЕ МАССИВЫ

Многомерный массив объявляется и создается следующим образом:

```
1  Int[][] myTwoDimentionalArray = new int [8][8];
```

Для работы с массивами в Java есть класс java.util.Arrays (arrays на английском и означает “массивы”). В целом с массивами чаще всего проделывают следующие операции: заполнение элементами (инициализация), извлечение элемента (по номеру), сортировка и поиск.

КОНТРОЛЬНАЯ РАБОТА

1. **Вариант 1**

   Напишите программу, которая находит максимальную последовательность последовательных равных элементов в массиве. Например: {1, 1, 2, 3, 2, 2, 2, 1} -> {2, 2, 2}

   Вариант №2

   Напишите программу, которая находит максимальную последовательность последовательно расположенных возрастающих целых чисел. Пример: {3, 2, 3, 4, 2, 2, 4} -> {2, 3, 4}.

2. **Вариант №1**

   Написать программу, которая выводит все комбинации (1*(2*(3*(4*5)))) = A,
   где вместо * - любая арифметическая операция +,-.*.
   A – натуральное число, которое вводится в программу.

   Вариант №2

   Напишите программу для поиска последовательности соседних чисел в массиве, которая имеет сумму определенного числа S. Пример: {4, 3, 1, 4, 2, 5, 8}, S = 11 -> {4, 2, 5}.